



(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
02.12.2020 Bulletin 2020/49

(51) Int Cl.:
H04L 29/08 (2006.01) H04L 12/26 (2006.01)

(21) Application number: **20172961.3**

(22) Date of filing: **09.03.2016**

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR

(72) Inventor: **MALPASS, Luke**
Stoke on Trent (GB)

(74) Representative: **Adamson, Katherine Louise**
Marks & Clerk LLP
40 Torphichen Street
Edinburgh EH3 8JB (GB)

(30) Priority: **11.03.2015 US 201562131619 P**

(62) Document number(s) of the earlier application(s) in accordance with Art. 76 EPC:
16762464.2 / 3 269 124

(71) Applicant: **Fasetto, Inc.**
Superior, WI 54880 (US)

Remarks:

This application was filed on 05-05-2020 as a divisional application to the application mentioned under INID code 62.

(54) **SYSTEMS AND METHODS FOR WEB API COMMUNICATION**

(57) A method of communication between devices is provided. The method comprises receiving a connection request, establishing a connection in response to the connection request, and receiving a first web API request on the connection. The method further comprises exe-

cuting a first procedure in response to the first web API request, receiving a second web API request on the connection, and executing at least one of the first procedure or a second procedure in response to the second web API request.

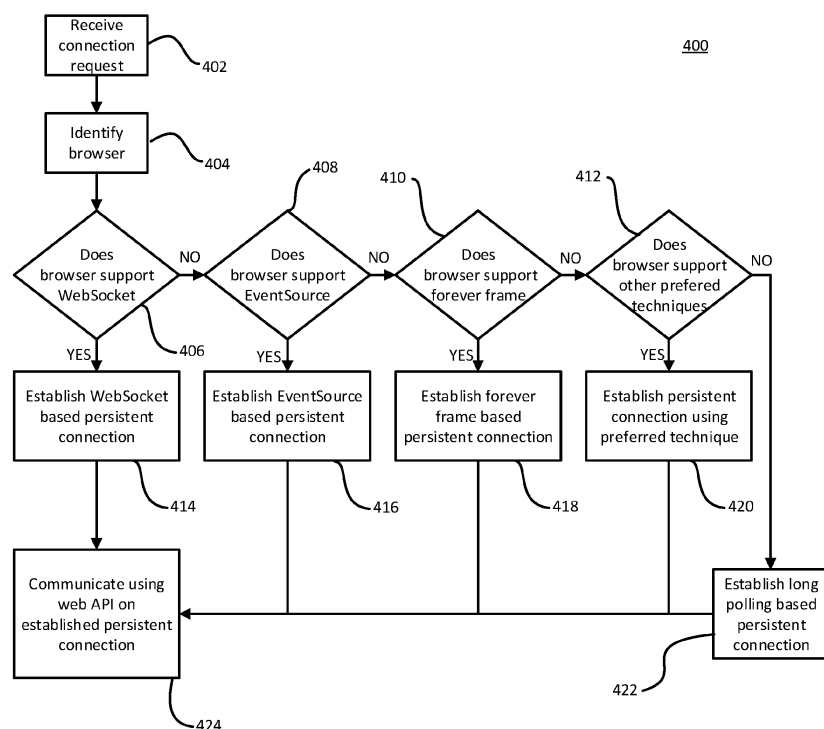


FIG. 4

Description

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This Application claims priority to U.S. Provisional Patent Application Serial No. 62/131,619, entitled "SYSTEMS AND METHODS FOR WEB API COMMUNICATION," and filed March 11, 2015, which is incorporated herein by reference in its entirety.

FIELD

[0002] The present disclosure generally relates to communication using web APIs, and more particularly to systems and methods for web API communication over a persistent connection. As used herein, the phrase "web API" refers to a collection of URLs (i.e., web addresses) made available to invoke a predefined set of procedures on a remote device.

BACKGROUND

[0003] Communication between devices is typically performed over a network, such as the internet or a local area network. Applications that run on devices may also be in communication with one another. One way that different applications can interact is through APIs. A web API, for example, may enable a third-party application to communicate with a server that hosts data for an existing application.

[0004] The web API may require an application to open and close a connection for each request sent to the server. In that regard, each request is a single, disconnected, and isolated call. The overhead associated with opening and closing a connection with each request may result in poor application performance. Communication between the third-party application and the server may appear slow to a client running the third-party application, for example. The appearance of slow communication and/or long load times may turn away consumers that would otherwise use an application.

SUMMARY

[0005] A method of communication between devices comprises the steps of receiving a connection request, establishing a connection in response to the connection request, and receiving a first web API request on the connection. The method further comprises the steps of executing a first procedure in response to the first web API request, receiving a second web API request on the connection, and executing at least one of the first procedure or a second procedure in response to the second web API request.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] A more complete understanding may be de-

rived by referring to the detailed description and claims when considered in connection with the Figures, wherein like reference numbers refer to similar elements throughout the Figures, and:

FIG. 1 illustrates a schematic diagram of a system for implementing a web API with a persistent connection according to various embodiments of the disclosure;

FIG. 2 illustrates a process for a server establishing a connection with a client and transmitting web API requests and responses between the server and client using a persistent connection according to various embodiments;

FIG. 3 illustrates a process for a client establishing a connection with a server and transmitting web API requests and responses between the server and client using a persistent connection according to various embodiments;

FIG. 4 illustrates a logic for selecting a type of persistent connection with a server to support an API over the connection according to various embodiments; and

FIG. 5 illustrates an API communication system between two communication devices over time according to various embodiments.

DETAILED DESCRIPTION

[0007] The detailed description of exemplary embodiments herein makes reference to the accompanying drawings and pictures, which show various embodiments by way of illustration. While these various embodiments are described in sufficient detail to enable those skilled in the art to practice the disclosure, it should be understood that other embodiments may be realized and that logical and mechanical changes may be made without departing from the spirit and scope of the disclosure. Thus, the detailed description herein is presented for purposes of illustration only and not of limitation. For example, the steps recited in any of the method or process descriptions may be executed in any order and are not limited to the order presented. Moreover, any of the functions or steps may be outsourced to or performed by one or more third parties. Furthermore, any reference to singular includes plural embodiments, and any reference to more than one component may include a singular embodiment.

[0008] Systems, methods and computer program products are provided. In the detailed description herein, references to "various embodiments," "one embodiment," "an embodiment," "an example embodiment," etc., indicate that the embodiment described may include a particular feature, structure, or characteristic, but every embodiment may not necessarily include the particular feature, structure, or characteristic. Moreover, such phrases are not necessarily referring to the same embodiment. Further, when a particular feature, structure,

or characteristic is described in connection with an embodiment, it is submitted that it is within the knowledge of one skilled in the art to effect such feature, structure, or characteristic in connection with other embodiments whether or not explicitly described. After reading the description, it will be apparent to one skilled in the relevant art(s) how to implement the disclosure in alternative embodiments.

[0009] Systems and methods are disclosed herein for communication between electronic communication devices. As used herein, a "communication device" may refer to any device capable of communication with another device. For, example and without limitation, a communication device may refer to a smartphone, PDA, laptop, desktop computer, portable phone, GPS device, car navigation system, wireless printers, or any other device.

[0010] Referring to FIG. 1, a system 100 for web application programming interface (API) communication over a persistent connection is illustrated according to various embodiments. System 100 includes a first communication device 102. An application 104 is running on first communication device 102. System 100 also includes a second communication device 108. Second communication device 108 may run a second application 106. First communication device 102 and second communication device 108 may be in communication via network 110. Application 104 may provide data to and/or use data provided by second application 106. Similarly, second application 106 may provide data to and/or use data provided by application 104. In that regard, application 104 and second application 106 may utilize web APIs to communicate with one another.

[0011] The procedures of the web API may be called by a first communication device 102 in communication with second communication device 108 over network 110. For example, a procedure to request a folder from second communication device 108 may look like `http://fakeurl.com/api/getfolder`. When second communication device 108 (e.g., a server) receives the request from first communication device 102 (e.g., a client), the communication device may respond by providing the requested data, in this case folder information, in a predetermined format (e.g. JSON or XML) to the first communication device 102.

[0012] In various embodiments, application 104 may use a web API to interface with second application 106. Application 104 running on first communication device 102 may send an API request through network 110 to second communication device 108 running second application 106. Second application 106 may receive and process the API request. In response to receiving the API request, second application 106 may execute a procedure corresponding to the particular API request received.

[0013] In order to send and receive API requests, as described in further detail below, a connection is established between first communication device 102 and second communication device 108. By maintaining a per-

sistent connection between first communication device 102 and second communication device 108, system 100 may complete API requests more quickly than a system that opens and closes a connection for every API request.

[0014] Referring to FIG. 2, a process 200 for handling web API requests sent by first communication device 102 and received by second communication device 108 is illustrated according to various embodiments. Application 104 running on first communication device 102 may be programmed to provide data or acquire data from second application 106 running on second communication device 108. The second communication device 108 may receive a connection from first communication device 102 (Step 202). Application 104 may be a web client, for example, that submits a connection request to second communication device 108 that is functioning as a server. In response to receiving the request, second communication device 108 may establish a connection with first communication device 102 (Step 204). Once established, the connection may be maintained indefinitely.

[0015] In various embodiments, second communication device 108 may maintain a number of active connections in excess of the limits normally inherent on such devices by context switching between active connections with data being sent and active connections that have been idle for a time. Second communication device 108 may switch active connections that have been idle into a data storage system such as a SQL database, a database farm, a hash table, or any other means of storing connection information. When information is received for an active connection that had been idle and is in storage, the active connection may be switch out of storage and into active memory.

[0016] Any databases discussed herein may include relational, hierarchical, graphical, or object-oriented structure and/or any other database configurations. Common database products that may be used to implement the databases include DB2 by IBM (Armonk, NY), various database products available from Oracle Corporation (Redwood Shores, CA), Microsoft Access or Microsoft SQL Server by Microsoft Corporation (Redmond, Washington), MySQL by MySQL AB (Uppsala, Sweden), or any other suitable database product. Moreover, the databases may be organized in any suitable manner, for example, as data tables or lookup tables. Each record may be a single file, a series of files, a linked series of data fields or any other data structure. Association of certain data may be accomplished through any desired data association technique such as those known or practiced in the art. For example, the association may be accomplished either manually or automatically. Automatic association techniques may include, for example, a database search, a database merge, GREP, AGREP, SQL, using a key field in the tables to speed searches, sequential searches through all the tables and files, sorting records in the file according to a known order to simplify lookup, and/or the like. The association step may be accomplished by a database merge function, for example,

using a "key field" in pre-selected databases or data sectors. Various database tuning steps are contemplated to optimize database performance. For example, frequently used files such as indexes may be placed on separate file systems to reduce In/Out ("I/O") bottlenecks.

[0017] One skilled in the art will also appreciate that, for security reasons, any databases, systems, devices, servers or other components of the system may consist of any combination thereof at a single location or at multiple locations, wherein each database or system includes any of various suitable security features, such as firewalls, access codes, encryption, decryption, compression, decompression, and/or the like.

[0018] Encryption may be performed by way of any of the techniques now available in the art or which may become available e.g., Twofish, RSA, El Gamal, Schorr signature, DSA, PGP, PKI, GPG (GnuPG), and symmetric and asymmetric cryptosystems.

[0019] In various embodiments, second communication device 108 may then receive a first API request via the established connection (Step 206). In response to receiving the API request, second communication device 108 returns data via the established connection to first communication device 102 (Step 208). After returning the data, second communication device 108 may maintain the established connection with first communication device 102 (Step 210). The connection may be maintained using various methods discussed in further detail below with reference to FIG. 4.

[0020] In various embodiments, second communication device 108 may receive a second web API request via the established connection (Step 212). The second web API request may be received without an additional connection being established between second communication device 108 and first communication device 102. In response to receiving the second web API request, second communication device 108 returns data via the established connection to first communication device 102 (Step 214).

[0021] Referring to FIG. 3, a process 300 for sending API requests from first communication device 102 to second communication device 108 is illustrated according to various embodiments. Application 104 running on first communication device 102 may be programmed to provide data or acquire data from second application 106 running on second communication device 108 by sending an API request to second communication device 108. First communication device 102 may send a connection request to second communication device 108 (Step 302). Application 104 may be a web client, for example, that submits a connection request to second communication device 108, which is functioning as a server. In response to receiving the request, second communication device 108 may establish a connection with first communication device 102 (Step 304). Second communication device 108 may then send a first API request via the established connection (Step 306). In response to receiving the API request, first communication device 102 receives data

via the established connection (Step 308). After receiving the data, first communication device 102 may maintain the established connection with second communication device 108 (Step 310). The connection may be maintained using various methods discussed in further detail below with reference to FIGs. 4 - 5.

[0022] In various embodiments, first communication device 102 may send a second web API request via the established connection (Step 312). The second web API request may be sent without establishing an additional connection between first communication device 102 and second communication device 108. In response to sending the second web API request, first communication device 102 receives data via the established connection from second communication device 108 (Step 314).

[0023] The methods of receiving web API requests (FIG. 2) and sending web API requests (FIG. 3) may be used, for example, to provide a web site with a nearly instantaneous response time. After the connection between first communication device 102 and second communication device 108 is established, subsequent web API calls no longer impose the overhead of establishing a connection. Thus, the result of the web API call is carried out with reduced latency. In that regard, web API calls using a persistent connection may appear seemingly instantaneous to users in many instances.

[0024] With reference to FIG. 4, a process 400 for establishing a persistent connection between first communication device 102 and second communication device 108 of FIG. 1 is shown. Second communication device 108 may be operating as a server while interacting with first communication device 102, which may be operating as a web client. Second communication device 108 may receive a connection request from first communication device 102 (Step 402). In response to the request, second communication device 108 may identify the application 104 (e.g., a browser) running on first communication device 102 that is making the connection request (Step 404). Second communication device 108 may identify the browser using device identifiers, operating system identifiers, and/or application identifiers, for example. A browser identifier and device identifier may be provided along with the connection request. For example, second communication device 108 may receive a browser name, browser version, operating system name, and operating version along with a connection request. Second communication device 108 may also request browser name, browser version, operating system name, and operating version in response to receiving a connection request, depending on the configuration of second communication device 108.

[0025] Once second communication device 108 has identifying information, second communication device 108 may check the identifying information to determine what type of connection to establish. Second communication device 108 may start by checking whether the identified browser supports WebSocket connections (Step 406). WebSocket is a protocol for providing bi-di-

rectional communication between first communication device 102 and second communication device 108. WebSocket connections may only be compatible with modern browsers that are coded to handle WebSocket connections. Although, WebSocket provides a clean, dedicated solution for persistent connections, many devices in use are not compatible with WebSocket connections. Thus, if application 104 running on first communication device 102 is compatible with WebSocket connections, second communication device 108 will establish a WebSocket based persistent connection with first communication device 102 (Step 414). If application 104 running on first communication device 102 is not compatible with WebSocket connections, then second communication device 108 may check for other compatible connection types.

[0026] The second communication device 108 may also check whether the identified browser supports EventSource (i.e., server-sent events) connections (Step 408). An EventSource connection also enables a server to push communications to a client, thus enabling bi-directional communication. EventSource connections may be compatible with a different list of browsers and devices than WebSocket connections. Thus, if application 104 running on first communication device 102 is compatible with EventSource, second communication device 108 will establish an EventSource based persistent connection with first communication device 102 (Step 416). If application 104 running on first communication device 102 is not compatible with EventSource connections, then second communication device 108 may check for other compatible connection types.

[0027] The second communication device 108 may then check whether the identified browser supports forever frame connections (Step 410). A forever frame connection is another technique to enable a server to push communications to a client, thus enabling bi-directional communication. The forever frame connection may be established by creating a hidden iframe element in a browser and establishing a connection inside the hidden frame. A forever frame connection may be compatible with a different list of browsers and devices than WebSocket and EventSource connections. Thus, if application 104 running on first communication device 102 is compatible with forever framing, second communication device 108 will establish a forever-frame-based persistent connection with first communication device 102 (Step 418). If application 104 running on first communication device 102 is not compatible with forever frame connections, then second communication device 108 may check for other compatible connection types.

[0028] In various embodiments, additional preferred techniques may be incorporated into the checks to see which connection technique second communication device 108 will use to establish a persistent connection. To that end, second communication device may check whether the identified browser supports another preferred connection technique (Step 412). If application 104 running on first communication device 102 is compatible

with a preferred connection technique, second communication device 108 will establish a persistent connection with first communication device 102 using the preferred technique (Step 420). If application 104 running on first communication device 102 is not compatible with a preferred connection technique, then second communication device 108 may use the default connection. The default connection as illustrated in FIG. 4 comprises establishing a long-polling based persistent connection (Step 422).

[0029] With brief reference to FIG. 1, once a persistent connection is established between first communication device 102 and second communication device 108, application 104 may communicate with second application 106 across network 110 with reduced overhead. Referring to FIG. 5, API communication system 500 is shown over time using a persistent connection between application 104 and second application 106, running on first communication device 102 and second communication device 108, respectively. As illustrated, time passes linearly as position on timeline T moves from left to right. Application 104 may send a request 502 for a connection to second application 106. Application 104 and second application 106 may complete the connection process with communication 504 from second application 106 to application 104 after time period P_0 has elapsed. A typical time period P_0 for establishing a connection may last from approximately 50 ms to 500 ms, depending on latency between communication devices.

[0030] After the connection is established, applications running on first communication device 102 and second communication device 108 may communicate using a web API over the established, persistent connection. Application 104 sends an API request 506 to second application 106. Second application 106 may locally execute a function corresponding to the API request 506 and then return corresponding data 508. The time period P_1 from sending the API request 506 to returning corresponding data 508 may be approximately 12 ms, for example. If the connection were closed within 10 ms after returning corresponding data 508 then the overhead for a single API communication between application 104 and second application 106 would be $P_0 + P_1 + 10$ ms. However, because a persistent connection has been established (as illustrated in FIG. 4, for example) the connection remains open after second application 106 returns corresponding data 508.

[0031] Second communication device 108 running application 106 may then push information 510 to application 104 over the established, persistent connection as information 510 becomes available. For example, application 104 may be a text-based chat application that sends and receives chat messages through second application 106. Information 510 may indicate that a user is typing a chat message directed to application 104. Second application 106 may provide information 510 to application 104 so that application 104 can display an indicator that the user is typing a message. Information 510

is passed to application 104 without application 104 directly requesting information 510. In that regard, API communication system 500 may eliminate the overhead that application 104 would otherwise incur to receive information 510 because application 104 can simply listen on an established, persistent connection rather than repeatedly establishing a new connection (incurring connection overhead time period P_0 each time) to request information 510.

[0032] Application 104 may then send a second API request 512 to second application 106 using API communication system 500 with an established, persistent connection. Application 104 may send second API request 512 without establishing a new connection (and without incurring overhead time period P_0 for establishing a connection). Second application 106 may process second API request 512 and return results 514 after time period P_2 . Time period P_2 may have duration of approximately 14 ms, for example. The connection may remain open during period P_4 , while waiting for further communication between application 104 and second application 106. Second communication device 108 may send a request 516 to application 104 for information over the established connection. In that regard, a server may communicate with a client by directly requesting information from the client. The client may then provide the requested information 518 to the server after period P_3 . Information requested by second communication device 108 might be geographic location or local time, for example. The persistent connection (as described with reference to FIGs. 2 through 4) enables near instantaneous communication between a server and a client by reducing the overhead ancillary to typical communications between server and client using an API. In particular, the persistent connection may reduce the time overhead for a given communication by at least the time period P_0 typically used to establish a connection.

[0033] In various embodiments, the methods described herein are implemented using the various particular machines described herein. The methods described herein may be implemented using the below particular machines, and those hereinafter developed, in any suitable combination, as would be appreciated immediately by one skilled in the art. Further, as is unambiguous from this disclosure, the methods described herein may result in various transformations of certain articles.

[0034] For the sake of brevity, conventional data networking, application development and other functional aspects of the systems (and components of the individual operating components of the systems) may not be described in detail herein. Furthermore, the connecting lines shown in the various figures contained herein are intended to represent exemplary functional relationships and/or physical couplings between the various elements. It should be noted that many alternative or additional functional relationships or physical connections may be present in a practical system.

[0035] The various system components discussed

herein may include one or more of the following: a host server or other computing systems including a processor for processing digital data; a memory coupled to the processor for storing digital data; an input digitizer coupled to the processor for inputting digital data; an application program stored in the memory and accessible by the processor for directing processing of digital data by the processor; a display device coupled to the processor and memory for displaying information derived from digital data processed by the processor; and a plurality of databases. Various databases used herein may include connection data and/or like data useful in the operation of the system. As those skilled in the art will appreciate, user computer may include an operating system (e.g., Windows NT, Windows 95/98/2000, Windows XP, Windows Vista, Windows 7, Windows 8, OS2, UNIX, Linux, Solaris, MacOS, etc.) as well as various conventional support software and drivers typically associated with computers.

[0036] The present system or any part(s) or function(s) thereof may be implemented using hardware, software or a combination thereof and may be implemented in one or more computer systems or other processing systems. However, the manipulations performed by embodiments were often referred to in terms, such as matching or selecting, which are commonly associated with mental operations performed by a human operator. No such capability of a human operator is necessary, or desirable in most cases, in any of the operations described herein. Rather, the operations may be machine operations. Useful machines for performing the various embodiments include general purpose digital computers or similar devices.

[0037] In fact, in various embodiments, the embodiments are directed toward one or more computer systems capable of carrying out the functionality described herein. The computer system includes one or more processors. The processor is connected to a communication infrastructure (e.g., a communications bus, cross over bar, or network). Various software embodiments are described in terms of this exemplary computer system. After reading this description, it will become apparent to a person skilled in the relevant art(s) how to implement various embodiments using other computer systems and/or architectures. Computer system can include a display interface that forwards graphics, text, and other data from the communication infrastructure (or from a frame buffer not shown) for display on a display unit.

[0038] Computer system also includes a main memory, such as for example random access memory (RAM), and may also include a secondary memory. The secondary memory may include, for example, a hard disk drive and/or a removable storage drive, representing a floppy disk drive, a magnetic tape drive, an optical disk drive, etc. The removable storage drive reads from and/or writes to a removable storage unit in a well-known manner. Removable storage unit represents a floppy disk, magnetic tape, optical disk, solid state disk, etc. which is

read by and written to by removable storage drive. As will be appreciated, the removable storage unit includes a computer usable storage medium having stored therein computer software and/or data.

[0039] In various embodiments, secondary memory may include other similar devices for allowing computer programs or other instructions to be loaded into computer system. Such devices may include, for example, a removable storage unit and an interface. Examples of such may include a program cartridge and cartridge interface (such as that found in video game devices), a removable memory chip (such as an erasable programmable read only memory (EPROM), or programmable read only memory (PROM)) and associated socket, and other removable storage units and interfaces, which allow software and data to be transferred from the removable storage unit to computer system.

[0040] Computer system may also include a communications interface. Communications interface allows software and data to be transferred between computer system and external devices. Examples of communications interface may include a modem, a network interface (such as an Ethernet card), a communications port, a Personal Computer Memory Card International Association (PCMCIA) slot and card, etc. Software and data transferred via communications interface are in the form of signals which may be electronic, electromagnetic, optical or other signals capable of being received by communications interface. These signals are provided to communications interface via a communications path (e.g., channel). This channel carries signals and may be implemented using wire, cable, fiber optics, a telephone line, a cellular link, a radio frequency (RF) link, wireless and other communications channels.

[0041] The terms "computer program medium" and "computer usable medium" are used to generally refer to media such as removable storage drive and a hard disk installed in hard disk drive. These computer program products provide software to computer system.

[0042] Computer programs (also referred to as computer control logic) are stored in main memory and/or secondary memory. Computer programs may also be received via communications interface. Such computer programs, when executed, enable the computer system to perform the features as discussed herein. In particular, the computer programs, when executed, enable the processor to perform the features of various embodiments. Accordingly, such computer programs represent controllers of the computer system.

[0043] In various embodiments, software may be stored in a computer program product and loaded into computer system using removable storage drive, hard disk drive or communications interface. The control logic (software), when executed by the processor, causes the processor to perform the functions of various embodiments as described herein. In various embodiments, hardware components such as application specific integrated circuits (ASICs). Implementation of the hardware

state machine so as to perform the functions described herein will be apparent to persons skilled in the relevant art(s).

[0044] In various embodiments, the server may include application servers (e.g. WEB SPHERE, WEB LOGIC, JBOSS). In various embodiments, the server may include web servers (e.g. APACHE, IIS, GWS, SUN JAVA SYSTEM WEB SERVER).

[0045] As those skilled in the art will appreciate, a device may include but is not limited to an operating system as well as various conventional support software and drivers typically associated with computers. A device may include but is not limited to any suitable personal computer, network computer, workstation, personal digital assistant, cellular phone, smart phone, minicomputer, mainframe or the like. A device can be in a home or business environment with access to a network. In various embodiments, access is through a network or the Internet through a commercially available web-browser software package. A device may implement security protocols such as Secure Sockets Layer (SSL) and Transport Layer Security (TLS). A device may implement several application layer protocols including http, https, ftp, and sftp.

[0046] In various embodiments, components, modules, and/or engines of system 100 may be implemented as micro-applications or micro-apps. Micro-apps are typically deployed in the context of a mobile operating system, including for example, a Palm mobile operating system, a Windows mobile operating system, an Android Operating System, Apple iOS, a Blackberry operating system and the like. The micro-app may be configured to leverage the resources of the larger operating system and associated hardware via a set of predetermined rules which govern the operations of various operating systems and hardware resources. For example, where a micro-app desires to communicate with a device or network other than the mobile device or mobile operating system, the micro-app may leverage the communication protocol of the operating system and associated device hardware under the predetermined rules of the mobile operating system. Moreover, where the micro-app desires an input from a user, the micro-app may be configured to request a response from the operating system which monitors various hardware components and then communicates a detected input from the hardware to the micro-app.

[0047] "Cloud" or "Cloud computing" includes a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. Cloud computing may include location-independent computing, whereby shared servers provide resources, software, and data to computers and other devices on demand. For more information regarding cloud computing, see the NIST's (National Institute of Standards and Technology) definition of cloud computing at <http://cs-crc.nist.gov/groups/SNS/cloud-computing/cloud-def->

vl5.doc (last visited February 4, 2011), which is hereby incorporated by reference in its entirety.

[0048] As used herein, "transmit" may include sending electronic data from one system component to another. Additionally, as used herein, "data" may include encompassing information such as commands, queries, files, data for storage, and the like in digital or any other form.

[0049] The system contemplates uses in association with web services, utility computing, pervasive and individualized computing, security and identity solutions, autonomic computing, cloud computing, commodity computing, mobility and wireless solutions, open source, biometrics, grid computing and/or mesh computing.

[0050] The computers discussed herein may provide a suitable website or other Internet-based graphical user interface which is accessible by users. In various embodiments, the Microsoft Internet Information Server (IIS), Microsoft Transaction Server (MTS), and Microsoft SQL Server, are used in conjunction with the Microsoft operating system, Microsoft NT web server software, a Microsoft SQL Server database system, and a Microsoft Commerce Server. Additionally, components such as Access or Microsoft SQL Server, Oracle, Sybase, Informix MySQL, Interbase, etc., may be used to provide an Active Data Object (ADO) compliant database management system. In various embodiments, the Apache web server is used in conjunction with a Linux operating system, a MySQL database, and the Perl, PHP, and/or Python programming languages.

[0051] Any of the communications, inputs, storage, databases or displays discussed herein may be facilitated through a website having web pages. The term "web page" as it is used herein is not meant to limit the type of documents and applications that might be used to interact with the user. For example, a typical website might include, in addition to standard HTML documents, various forms, Java applets, JavaScript, active server pages (ASP), common gateway interface scripts (CGI), extensible markup language (XML), dynamic HTML, cascading style sheets (CSS), AJAX (Asynchronous Javascript And XML), helper applications, plug-ins, and the like. A server may include a web service that receives a request from a web server, the request including a URL (<http://yahoo.com/stockquotes/ge>) and an IP address (123.56.789.234). The web server retrieves the appropriate web pages and sends the data or applications for the web pages to the IP address. Web services are applications that are capable of interacting with other applications over a communications means, such as the internet. Web services are typically based on standards or protocols such as XML, SOAP, AJAX, WSDL and UDDI. Web services methods are well known in the art, and are covered in many standard texts. See, e.g., ALEX NGHIEM, IT WEB SERVICES: A ROADMAP FOR THE ENTERPRISE (2003), hereby incorporated by reference.

[0052] Practitioners will also appreciate that there are a number of methods for displaying data within a browser-based document. Data may be represented as standard

text or within a fixed list, scrollable list, drop-down list, editable text field, fixed text field, pop-up window, and the like. Likewise, there are a number of methods available for modifying data in a web page such as, for example, free text entry using a keyboard, selection of menu items, check boxes, option boxes, and the like.

[0053] The system and method may be described herein in terms of functional block components, screen shots, optional selections and various processing steps. It should be appreciated that such functional blocks may be realized by any number of hardware and/or software components configured to perform the specified functions. For example, the system may employ various integrated circuit components, e.g., memory elements, processing elements, logic elements, look-up tables, and the like, which may carry out a variety of functions under the control of one or more microprocessors or other control devices. Similarly, the software elements of the system may be implemented with any programming or scripting language such as C, C++, C#, Java, JavaScript, VBScript, Macromedia Cold Fusion, COBOL, Microsoft Active Server Pages, assembly, PERL, PHP, awk, Python, Visual Basic, SQL Stored Procedures, PL/SQL, any UNIX shell script, and extensible markup language (XML) with the various algorithms being implemented with any combination of data structures, objects, processes, routines or other programming elements. Further, it should be noted that the system may employ any number of conventional techniques for data transmission, signaling, data processing, network control, and the like. Still further, the system could be used to detect or prevent security issues with a client-side scripting language, such as JavaScript, VBScript or the like. For a basic introduction of cryptography and network security, see any of the following references: (1) "Applied Cryptography: Protocols, Algorithms, And Source Code In C," by Bruce Schneier, published by John Wiley & Sons (second edition, 1995); (2) "Java Cryptography" by Jonathan Knudson, published by O'Reilly & Associates (1998); (3) "Cryptography & Network Security: Principles & Practice" by William Stallings, published by Prentice Hall; all of which are hereby incorporated by reference.

[0054] As will be appreciated by one of ordinary skill in the art, the system may be embodied as a customization of an existing system, an add-on product, a processing apparatus executing upgraded software, a standalone system, a distributed system, a method, a data processing system, a device for data processing, and/or a computer program product. Accordingly, any portion of the system or a module may take the form of a processing apparatus executing code, an internet based embodiment, an entirely hardware embodiment, or an embodiment combining aspects of the internet, software and hardware. Furthermore, the system may take the form of a computer program product on a computer-readable storage medium having computer-readable program code means embodied in the storage medium. Any suitable computer-readable storage medium may be utilized,

including hard disks, CD-ROM, optical storage devices, magnetic storage devices, solid state storage devices, and/or the like.

[0055] The system and method is described herein with reference to screen shots, block diagrams and flowchart illustrations of methods, apparatus (e.g., systems), and computer program products according to various embodiments. It will be understood that each functional block of the block diagrams and the flowchart illustrations, and combinations of functional blocks in the block diagrams and flowchart illustrations, respectively, can be implemented by computer program instructions.

[0056] These computer program instructions may be loaded onto a general purpose computer, special purpose computer, or other programmable data processing apparatus to produce a machine, such that the instructions that execute on the computer or other programmable data processing apparatus create means for implementing the functions specified in the flowchart block or blocks. These computer program instructions may also be stored in a computer-readable memory that can direct a computer or other programmable data processing apparatus to function in a particular manner, such that the instructions stored in the computer-readable memory produce an article of manufacture including instruction means which implement the function specified in the flowchart block or blocks. The computer program instructions may also be loaded onto a computer or other programmable data processing apparatus to cause a series of operational steps to be performed on the computer or other programmable apparatus to produce a computer-implemented process such that the instructions which execute on the computer or other programmable apparatus provide steps for implementing the functions specified in the flowchart block or blocks.

[0057] Accordingly, functional blocks of the block diagrams and flowchart illustrations support combinations of means for performing the specified functions, combinations of steps for performing the specified functions, and program instruction means for performing the specified functions. It will also be understood that each functional block of the block diagrams and flowchart illustrations, and combinations of functional blocks in the block diagrams and flowchart illustrations, can be implemented by either special purpose hardware-based computer systems which perform the specified functions or steps, or suitable combinations of special purpose hardware and computer instructions. Further, illustrations of the process flows and the descriptions thereof may make reference to user windows, webpages, websites, web forms, prompts, etc. Practitioners will appreciate that the illustrated steps described herein may comprise in any number of configurations including the use of windows, webpages, web forms, popup windows, prompts and the like. It should be further appreciated that the multiple steps as illustrated and described may be combined into single webpages and/or windows but have been expanded for the sake of simplicity. In other cases, steps illus-

trated and described as single process steps may be separated into multiple webpages and/or windows but have been combined for simplicity.

[0058] The term "non-transitory" is to be understood to remove only propagating transitory signals per se from the claim scope and does not relinquish rights to all standard computer-readable media that are not only propagating transitory signals per se. Stated another way, the meaning of the term "non-transitory computer-readable medium" and "non-transitory computer-readable storage medium" should be construed to exclude only those types of transitory computer-readable media which were found in *In Re Nuijten* to fall outside the scope of patentable subject matter under 35 U.S.C. § 101.

[0059] Benefits, other advantages, and solutions to problems have been described herein with regard to specific embodiments. However, the benefits, advantages, solutions to problems, and any elements that may cause any benefit, advantage, or solution to occur or become more pronounced are not to be construed as critical, required, or essential features or elements of the disclosure. Reference to an element in the singular is not intended to mean "one and only one" unless explicitly so stated, but rather "one or more." Moreover, where a phrase similar to 'at least one of A, B, and C' or 'at least one of A, B, or C' is used in the claims or specification, it is intended that the phrase be interpreted to mean that A alone may be present in an embodiment, B alone may be present in an embodiment, C alone may be present in an embodiment, or that any combination of the elements A, B and C may be present in a single embodiment; for example, A and B, A and C, B and C, or A and B and C.

[0060] Although the disclosure includes a method, it is contemplated that it may be embodied as computer program instructions on a tangible computer-readable carrier, such as a magnetic or optical memory or a magnetic or optical disk. All structural, chemical, and functional equivalents to the elements of the above-described exemplary embodiments that are known to those of ordinary skill in the art are expressly incorporated herein by reference and are intended to be encompassed by the present claims. Moreover, it is not necessary for a device or method to address each and every problem sought to be solved by the present disclosure, for it to be encompassed by the present claims. Furthermore, no element, component, or method step in the present disclosure is intended to be dedicated to the public regardless of whether the element, component, or method step is explicitly recited in the claims. No claim element herein is to be construed under the provisions of 35 U.S.C. 112(f) unless the element is expressly recited using the phrase "means for." As used herein, the terms "comprises", "comprising", or any other variation thereof, are intended to cover a non-exclusive inclusion, such that a process, method, article, or apparatus that comprises a list of elements does not include only those elements but may include other elements not expressly listed or inherent to such process, method, article, or apparatus.

[0061] Further embodiments are in accordance with the following numbered clauses:

- Clause 1. A method of communication between devices, comprising: 5
- receiving, by a server, a connection request;
 - establishing, by the server, a connection in response to the connection request;
 - receiving, by the server, a first web API request on the connection; 10
 - executing, by the server, a first procedure in response to the first web API request;
 - receiving, by the server, a second web API request on the connection; and 15
 - executing, by the server, at least one of the first procedure or a second procedure in response to the second web API request.
- Clause 2. The method of clause 1, wherein a type of the connection comprises at least one of a WebSocket based connection, an EventSource based connection, a forever frame based connection, or a long polling persistent connection. 20
- Clause 3. The method of clause 2, wherein the type of the connection is selected by the server based at least in part on compatibility of the type of the connection with a device transmitting the connection request. 25
- Clause 4. The method of clause 2, wherein the connection persists between the first web API request and the second web API request. 30
- Clause 5. The method of clause 4, wherein the connection persists after the second web API request. 35
- Clause 6. The method of clause 1, further comprising maintaining, by the server, the connection in response to executing the first procedure. 40
- Clause 7. The method of clause 1, further comprising switching, by the server, the connection into a data storage system in response to the connection being idle for a predetermined duration. 45
- Clause 8. The method of clause 7, further comprising switching, by the server, the connection into active memory in response receiving the second web API request on the connection. 50
- Clause 9. The method of clause 7, wherein the data storage system comprises at least one of a database, a database farm, a hash table, or a lookup table. 55
- Clause 10. A computer-based system, comprising:

a processor;
a tangible, non-transitory memory configured to communicate with the processor, the tangible, non-transitory memory having instructions stored thereon that, in response to execution by the processor, cause the processor to perform operations comprising:

- receiving, by the processor, a connection request from a communication device;
- establishing, by the processor, a connection to the communication device in response to the connection request;
- receiving, by the processor, a first web API request from the communication device via the connection;
- executing, by the processor, a first procedure in response to the first web API request;
- returning, by the processor, a result from the first procedure to the communication device via the connection;
- maintaining, by the processor, the connection in an open state in response to returning the result from the first procedure to the communication device;
- receiving, by the processor, a second web API request from the communication device on the connection; and
- executing, by the processor, at least one of the first procedure or a second procedure in response to the second web API request.

Clause 11. The computer-based system of clause 10, wherein a type of the connection comprises at least one of a WebSocket based connection, an EventSource based connection, a forever frame based connection, or a long polling based connection.

Clause 12. The computer-based system of clause 11, further comprising selecting, by the processor, the type of the connection based at least in part on compatibility of an application running on the communication device with the type of the connection.

Clause 13. The computer-based system of clause 11, wherein the connection persists between the first web API request and the second web API request.

Clause 14. The computer-based system of clause 13, wherein the connection persists after the second web API request.

Clause 15. The computer-based system of clause 10, further comprising switching, by the processor, the connection into a data storage system in response to the connection being idle for a predetermined duration.

mined duration.

Clause 16. The computer-based system of clause 15, further comprising switching, by the processor, the connection into active memory in response receiving the second web API request on the connection.

Clause 17. The computer-based system of clause 15, wherein the data storage system comprises at least one of a database, a database farm, a hash table, or a lookup table.

Clause 18. An article of manufacture including a non-transitory, tangible computer readable storage medium having instructions stored thereon that, in response to execution by a processor, cause the processor to perform operations comprising:

receiving, by the processor, a connection request from a communication device;
 establishing, by the processor, a connection to the communication device in response to the connection request;
 receiving, by the processor, a first web API request from the communication device via the connection;
 executing, by the processor, a first procedure in response to the first web API request;
 maintaining, by the processor, the connection in an open state in response to executing the first procedure; receiving, by the processor, a second web API request from the communication device on the connection in the open state; and
 executing, by the processor, at least one of the first procedure or a second procedure in response to the second web API request.

Clause 19. The article of clause 18, wherein a type of the connection comprises at least one of a WebSocket based connection, an EventSource based connection, a forever frame based connection, or a long polling based connection.

Clause 20. The article of clause 19, further comprising selecting, by the processor, the type of the connection based at least in part on compatibility of an application running on the communication device with the type of the connection.

Clause 21. The article of clause 19, wherein the connection persists between the first web API request and the second web API request.

Clause 22. The article of clause 21, wherein the connection persists after the second web API request.

Clause 23. The article of clause 18, further compris-

ing switching, by the processor, the connection into a data storage system in response to the connection being idle for a predetermined duration.

Clause 24. The article of clause 23, further comprising switching, by the processor, the connection into active memory in response receiving the second web API request on the connection.

Clause 25. The article of clause 23, wherein the data storage system comprises at least one of a database, a database farm, a hash table, or a lookup table.

Claims

1. A method of communication between devices, comprising:

receiving, by a server, a connection request from a first device;
 determining, by the server, that the first device is incompatible with a first type of connection;
 determining, by the server and in response to the first device being incompatible with the first type of connection, that the first device is compatible with a second type of connection, the second type of connection comprising at least one of a WebSocket based connection, an EventSource based connection, a forever frame based connection, or a long polling persistent connection;
 establishing, by the server, a connection of the second type of connection in response to the connection request; receiving, by the server, a first request on the connection;
 executing, by the server, a first procedure in response to the first request;
 receiving, by the server, a second request on the connection; and
 executing, by the server, at least one of the first procedure or a second procedure in response to the second request.

2. The method of claim 1, wherein the second type of connection comprises the EventSource based connection.
3. The method of claim 2, wherein the second type of connection is selected by the server based at least in part on compatibility of the second type of connection with the first device transmitting the connection request.
4. The method of claim 2, wherein the connection persists between the first request and the second request.

5. The method of claim 4, wherein the connection persists after the second request.
 6. The method of claim 1, further comprising maintaining, by the server, the connection in response to executing the first procedure. 5
 7. The method of claim 1, further comprising switching, by the server, the connection into a data storage system in response to the connection being idle for a predetermined duration. 10
 8. The method of claim 7, further comprising switching, by the server, the connection into active memory in response receiving the second request on the connection. 15
 9. The method of claim 7, wherein the data storage system comprises at least one of a database, a database farm, a hash table, or a lookup table. 20
 10. A computer-based system, comprising:
 - a processor;
 - a tangible, non-transitory memory configured to communicate with the processor, the tangible, non-transitory memory having instructions stored thereon that, in response to execution by the processor, cause the processor to perform operations comprising: 25
 - receiving, by the processor, a connection request from a communication device;
 - determining, by the processor, that the communication device is incompatible with a first type of connection;
 - determining, by the processor and in response to the communication device being incompatible with the first type of connection, that the communication device is compatible with a second type of connection, the second type of connection comprising at least one of WebSocket based connection, an EventSource based connection, a forever frame based connection, or a long polling persistent connection; 35
 - establishing, by the processor, a connection to the communication device in the second type of connection in response to the connection request; 40
 - receiving, by the processor, a first request from the communication device via the connection; 50
 - executing, by the processor, a first procedure in response to the first request; 55
 - returning, by the processor, a result from the first procedure to the communication device via the connection;
- maintaining, by the processor, the connection in an open state in response to returning the result from the first procedure to the communication device;
- receiving, by the processor, a second request from the communication device on the connection; and
- executing, by the processor, at least one of the first procedure or a second procedure in response to the second request.
11. The computer-based system of claim 10, wherein the second type of connection comprises the EventSource based connection.
 12. The computer-based system of claim 10 or 11, wherein at least one of a) to f):
 - a) the operations performed by the processor further comprises selecting, by the processor, the second type of connection based at least in part on compatibility of an application running on the communication device with the second type of connection;
 - b) the connection persists between the first request and the second request;
 - c) the connection persists after the second request;
 - d) the operations performed by the processor further comprise switching, by the processor, the connection into a data storage system in response to the connection being idle for a predetermined duration;
 - e) the operations performed by the processor further comprise switching, by the processor, the connection into active memory in response receiving the second request on the connection;
 - f) the data storage system comprises at least one of a database, a database farm, a hash table, or a lookup table.
 13. An article of manufacture including a non-transitory, tangible computer readable storage medium having instructions stored thereon that, in response to execution by a processor, cause the processor to perform operations comprising:
 - receiving, by the processor, a connection request from a communication device;
 - determining, by the processor, that the communication device is incompatible with a first type of connection;
 - determining, by the processor and in response to the communication device being incompatible with the first type of connection, that the communication device is compatible with a second type of connection, the second type of connection comprising at least one of a WebSocket

based connection an EventSource based connection, a forever frame based connection, or a long polling persistent connection;
 establishing, by the processor, a connection to the communication device in the second type of connection in response to the connection request;
 receiving, by the processor, a first request from the communication device via the connection;
 executing, by the processor, a first procedure in response to the first request;
 maintaining, by the processor, the connection in an open state in response to executing the first procedure;
 receiving, by the processor, a second request from the communication device on the connection in the open state; and
 executing, by the processor, at least one of the first procedure or a second procedure in response to the second request.

14. The article of claim 13, wherein the second type of connection comprises the EventSource based connection.

15. The article of claim 13 or 14, wherein at least one of a) to f):

a) the operations performed by the processor further comprise selecting, by the processor, the second type of connection based at least in part on compatibility of an application running on the communication device with the second type of connection;
 b) the connection persists between the first request and the second request;
 c) the connection persists after the second request;
 d) the operations performed by the processor further comprise switching, by the processor, the connection into a data storage system in response to the connection being idle for a predetermined duration;
 e) the operations performed by the processor further comprise switching, by the processor, the connection into active memory in response receiving the second request on the connection;
 f) the data storage system comprises at least one of a database, a database farm, a hash table, or a lookup table.

55

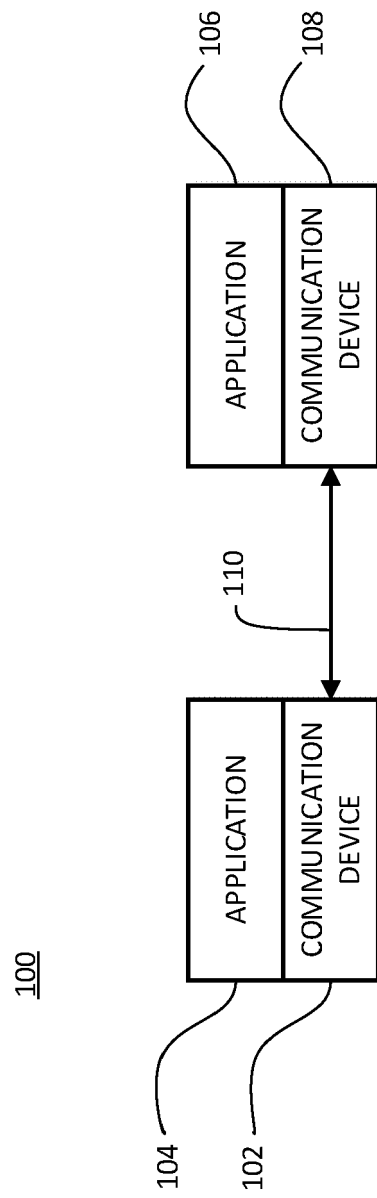


FIG. 1

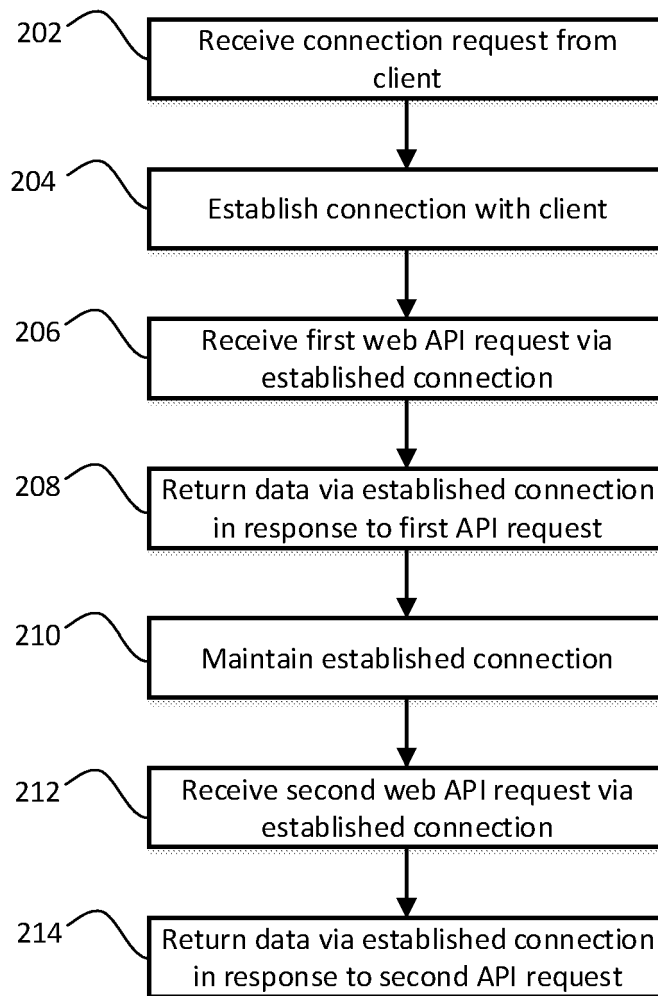
200

FIG. 2

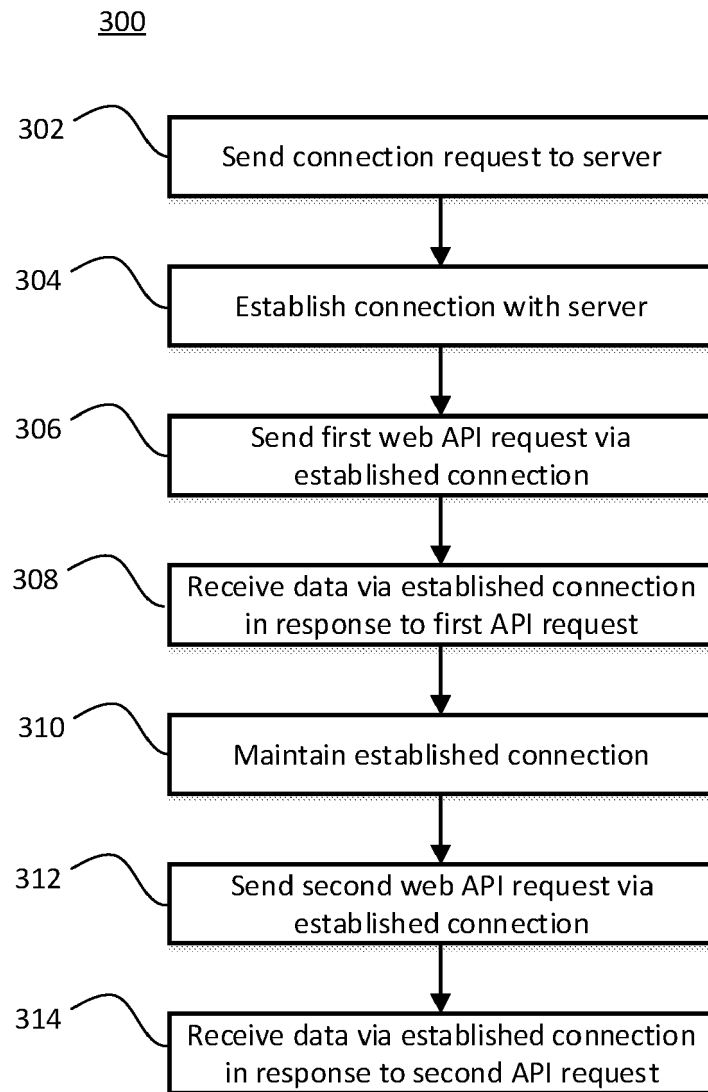


FIG. 3

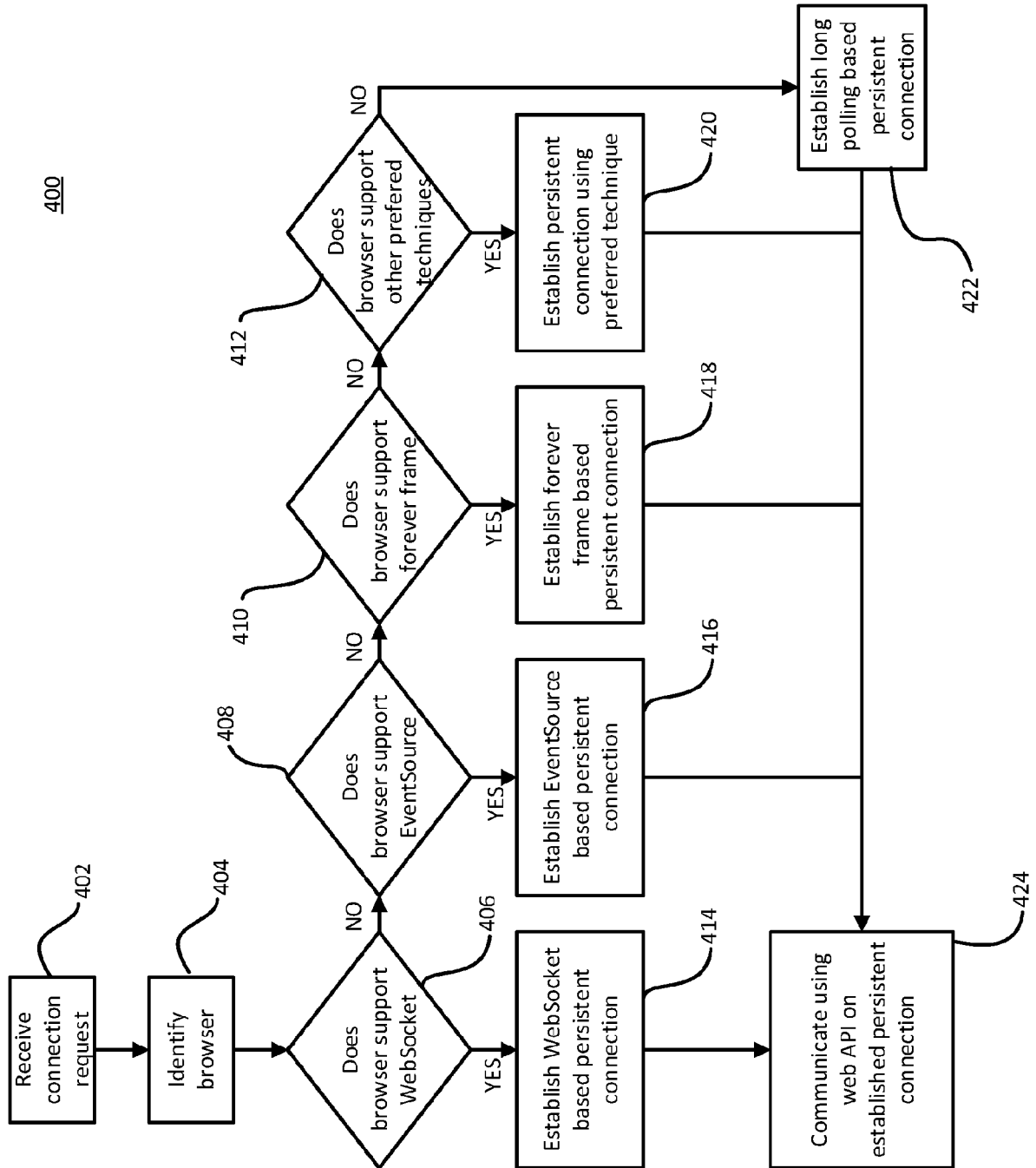


FIG. 4

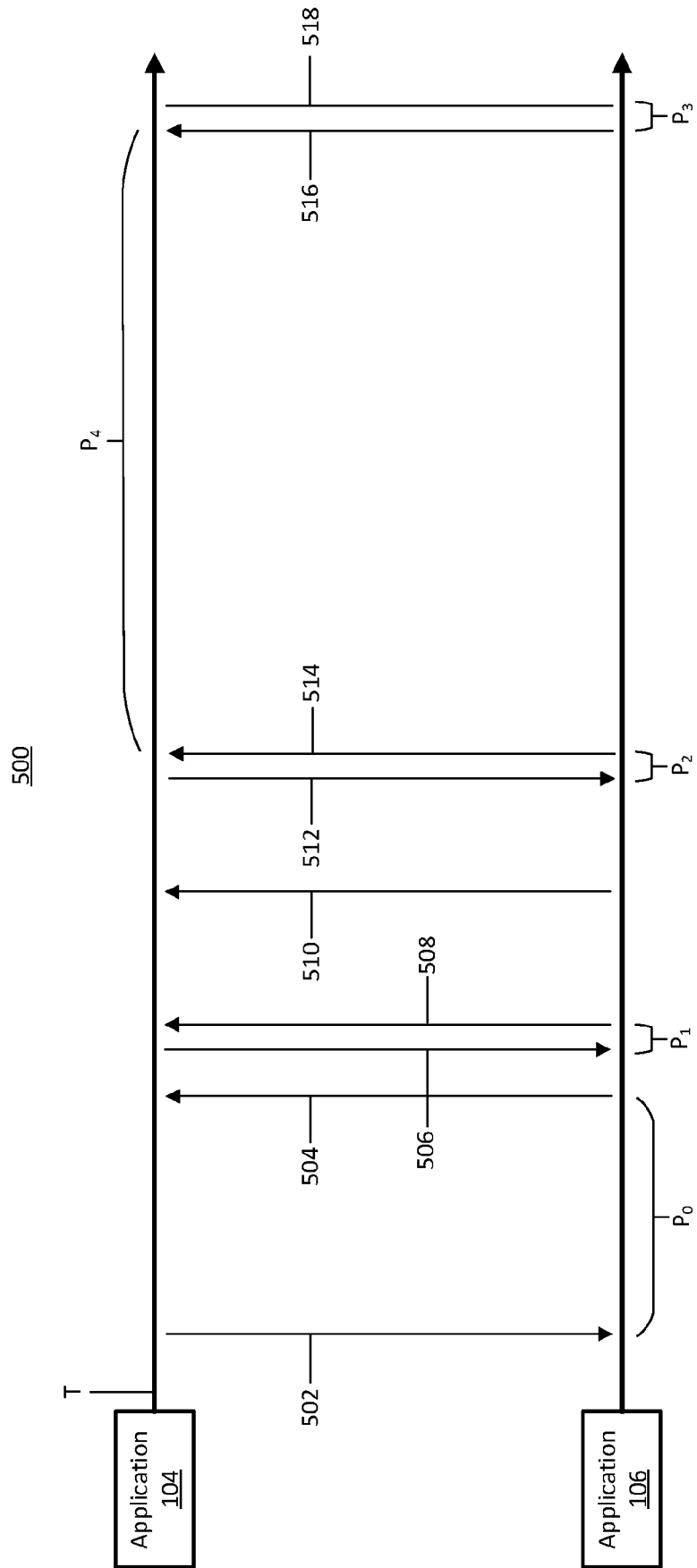


FIG. 5



EUROPEAN SEARCH REPORT

Application Number
EP 20 17 2961

5

10

15

20

25

30

35

40

45

50

55

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
X	CN 103 945 003 A (UNIV BEIJING POSTS & TELECOMM) 23 July 2014 (2014-07-23)	1-6, 10-15	INV. H04L29/08 H04L12/26
Y	* abstract * * paragraph [0009] - paragraph [0010] * * paragraph [0057] - paragraph [0059] * * paragraph [0068] * * figure 2 *	7-9	
X	US 2014/123033 A1 (UHMA GAVIN ANDREW ROSS [CA] ET AL) 1 May 2014 (2014-05-01)	1-6, 10-15	
Y	* paragraph [0037] - paragraph [0042] *	7-9	
Y	Anonymous: "PersistentManager (Catalina Internal API Documentation)", 10 November 2013 (2013-11-10), XP055514631, Retrieved from the Internet: URL:https://web.archive.org/web/20131110042918/https://tomcat.apache.org/tomcat-4.1-doc/catalina/docs/api/org/apache/catalina/session/PersistentManager.html [retrieved on 2018-10-11]	7-9	
A	* the whole document *	1-6, 10-15	TECHNICAL FIELDS SEARCHED (IPC) H04L
The present search report has been drawn up for all claims			
Place of search The Hague		Date of completion of the search 1 October 2020	Examiner Oechsner, Simon
CATEGORY OF CITED DOCUMENTS X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category A : technological background O : non-written disclosure P : intermediate document		T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application L : document cited for other reasons & : member of the same patent family, corresponding document	

EPO FORM 1503 03.82 (P04C01)

**ANNEX TO THE EUROPEAN SEARCH REPORT
ON EUROPEAN PATENT APPLICATION NO.**

EP 20 17 2961

5 This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report.
The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

01-10-2020

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
CN 103945003 A	23-07-2014	NONE	
US 2014123033 A1	01-05-2014	US 2012210243 A1	16-08-2012
		US 2014123033 A1	01-05-2014
		US 2015058407 A1	26-02-2015

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- US 62131619 [0001]

Non-patent literature cited in the description

- **BRUCE SCHNEIER.** Applied Cryptography: Protocols, Algorithms, And Source Code In C. John Wiley & Sons, 1995 [0053]
- **JONATHAN KNUDSON.** Java Cryptography. O'Reilly & Associates, 1998 [0053]
- **WILLIAM STALLINGS.** Cryptography & Network Security: Principles & Practice. Prentice Hall [0053]